EXCURSIONS TO A TOPOLOGICAL PICTUREBOOK

Ahmet Arslan and Varol Akman

BILKENT UNIVERSITY

**Department of Computer Engineering
and
Information Science**

Technical Report BU-CEIS-94-14

Excursions to a topological picturebook

Ahmet Arslan[1] and Varol Akman[2]

George K. Francis' A Topological Picturebook [Springer-Verlag, New York (1987)] is full of complicated topological figures, mostly drawn manually. Our goal is to automate the production of such illustrations and to obtain publication-quality hardcopy using assorted techniques of computer graphics. To that end, sweeping is discussed as a major surface modelling tool. Some interactive methods are given to produce interesting topological surfaces. The program T♭ *(which stands for '*Topology♭ook*') is described and various pictures generated by this software are presented.* T♭ *is a free-form surface modeller and produces topological shapes with little effort. Central to the implementation of* T♭ *is a paradigm of solid modelling in which computation of a shape is regarded as sweeping with some parametric variations, viz.* shape = sweep + control.

sweeping, surfaces, topology, solid modelling, B-spline curves, Bézier curves

A *free-form surface modeller* offers a set of surface design tools of which one can distinguish three types: *constructors*, *modifiers*, and *combiners* [14]. Constructors embed several methods for generating 3-D models of (real or imaginary) objects. The most popular approach is to use polygons as low-level building blocks which define increasingly complex objects. For example, thousands of polygons might be used to approximate a wine glass. Obviously, it is difficult and time consuming for a designer to define an object in this way. Therefore, a constructor must include higher level capabilities such as the description of a surface by interactive positioning of the control points or by a surface-fitting operator using interpolation. One objective of this class of tools is to allow the desired surfaces to be designed as simply as possible [18]. Hence, a capability to define generic constituents described by a few parameters is very useful. The constituents can be ordinary (a sphere, a torus) or more complicated (a surface of revolution). Sometimes it is necessary to use a nontrivial component such as a B-spline (or a Bézier) surface. Such surfaces or parametric patches require numerous calculations when it comes to rendering, although they can be represented in a compact form [8, 9].

Modifiers change the shape of an existing surface. The simplest technique is to move (or add) some control points, but more advanced operations have also been developed. For example, tapering, twisting, and blending techniques are available for smoothing and deforming a surface according to continuity constraints [23]. Finally, combiners are Boolean operators applied to the surfaces designed via constructors and modifiers. Each of these three operators has proved beneficial and therefore a modelling system should provide a general, accurate, and user-friendly framework incorporating them.

In this paper, the suggested techniques for modelling 3-D shapes are all based on sweeping. We give interactive methods for specifying and manipulating 3-D representations and discuss the desirable functionalities of an interactive graphics system to visualise the complicated shapes of low dimensional topology. The implemented program T♭ ('Topology♭ook') is a rudimentary graphical workbench to help topologists (also artists,

[1]Department of Electrical and Electronics Engineering, Fırat University, Elazığ, Turkey
[2]Department of Computer Engineering and Information Science, Bilkent University, Bilkent, Ankara 06533, Turkey

solid modellers, publishers, and geographers) to illustrate their ideas more effectively. T♭ was implemented in the C programming language and runs on a colour Sun workstation under the Sunview user interface system. The abbreviation T♭ is written in this way because we hope that our Topology♭ook combines truth (T) with beauty (indicated by the musical notation ♭).

Our research owes its existence to *A Topological Picturebook* [15] of George K. Francis—a book that was written to encourage topologists to illustrate their work and to help artists to understand the abstract (and invariably difficult) ideas expressed by such drawings. The reader is referred to references [2, 3] for early origins of the present work (albeit from a primarily geometrical viewpoint) and related ideas. Other relevant contributions on the theme of this paper can be found in references [10, 22].

Indulgence in a topological 'picturebook' may sound somewhat futile vis-à-vis the trend that sketching and visual presentation of topological constructions (and proofs) are slowly losing their stronghold which they once held. It may appear that the science of the 'deformation of shapes' is becoming sterile in terms of figures. However, we believe that there is definite place for a topological picturebook of the sort to be described here. The following excerpt Francis' book (p. 78) succinctly explains our view:

> [T]he pedagogy that underlies the entire book, and which I bring out specially here, comes from Bernard Morin of Strasbourg. Pictures without formulas mislead, formulas without pictures confuse. I don't know if Bernard would say it this way, but it is how I have understood his work ... Morin's vivid, pictorial description of his bold constructions has inspired their realisation in many a drawing, model, computer graphic, and film. But he insists that ultimately, pictorial descriptions should also be clothed in the analytical garb of traditional mathematics.

The outline of the paper is as follows. In the next section, a brief topological review will be presented. This is followed by a section wherein sweeping is illustrated and a new mathematical model of general, profiled, depth-modulated, and twisted-profiled sweep objects is given. Next, the implementation of T♭ is illustrated in a long section. Here, the user interface system of T♭ and modelling, editing, and local/global deformations of sweep objects are presented, along with ruled curves, skinning, and blending. Handle normalisation, shading, transparent windows on objects, transformations, text and image operations are also covered. Finally, a summary is offered and some colour pictures from T♭ are given.

TOPOLOGICAL REVIEW OF T♭

In this section, we briefly compare symbolic forms of some polyhedra in topology and their curve representations in T♭. The reader will find in the first subsection coverage of the standard representation of 2-manifolds. Essentially, the transformation of a surface from a symbolic form to a visual representation is where we use sweeping as a modeling technique. This point will be made clearer in the last subsection.

Normal forms for surfaces

Suppose we are given a surface made of rubber. If we distort it in a continuous manner without tearing it, the resulting surface will have the same chromatic number. Similarly, if a graph (which we may imagine to be made of wire) is distorted without tearing, many of the properties of the graph remain unchanged [16, 24]. The study of those properties of a figure which do not change after deformation is a main concern of topology. It is useful to give a rough idea about the kind of deformation we have in mind, viz. a homeomorphism. Some examples of pairs of homeomorphic figures are as follows:

- A circular disk, a square.
- The surface of a sphere, the surface of a tetrahedron.
- The interior of a circle, the interior of a rectangle.
- A set of n distinct points, another set of n distinct points.
- The surface of a torus, the surface of a tea cup.
- A solid torus, a tea cup.

A *circular disk* is the surface (the interior and the circumference) of a circle. Each topological image (in 3-space) of a circular disk is called a *2-cell*. Similarly, the topological image of an edge is called a *1-cell*. Consequently, a *0-cell* is just a point. By the term 'polygon' (triangle, quadrilateral, pentagon, hexagon, etc.) we do not restrict ourselves to plane figures. We define a *polygon* with r sides (an r-gon) as a 2-cell which has its circumference divided into r arcs by r vertices. The arcs are the *sides* of the polygon.

Given a finite number of polygons, let the total number of all of the sides of the polygons be even. These sides are given in pairs and are labeled. Two sides of a pair are labeled with the same letter. It is furthermore supposed that for each side an orientation is given (say, indicated by an arrow). Identifying each pair of sides such that the arrows coincide (i.e., point in the same direction) if one obtains a connected pseudograph consisting of the vertices and the edges (superimposed sides), then the figure so obtained is a *polyhedron*. The *faces* of the polyhedron are defined as usual.

The *plane representation* of a polyhedron is all the information that have been mentioned so far: the polygons, the pairing of the sides, and the orientation of the sides (before identification). A symbolic description of a polyhedron is created in the following way. For each polygon in the plane representation, choose an orientation. Then, for each polygon, write down the cyclic order of the sides as a row of the corresponding labels (put an inverse sign after the label if the orientation of the polyhedron is opposite to the orientation of the side). As a result a *scheme* (denoted by Σ)

$$\begin{array}{c} a \quad b \quad c^{-1} \quad \ldots \\ \ldots\ldots\ldots\ldots\ldots \\ \ldots\ldots\ldots\ldots\ldots \end{array}$$

is obtained with the following properties: (i) Each letter appears exactly twice in Σ, and (ii) The set of the rows in Σ cannot be separated into two disjoint subsets such that the property (i) holds for each of these subsets. The following operations change Σ but not the polyhedron it represents: (i) Put the first letter of a row behind the last letter, (ii) Whenever a certain label appears, say a, replace it with a^{-1} (and a^{-1} with a), and (iii) Simultaneously change all the signs in a row and reverse the cyclic order.

Given a polyhedron P and its symbolic representation scheme Σ, four *elementary operations* are considered. The property of these operations is that although they change P as well as Σ, the surface itself is not changed:

- SU1 (Subdivision of dimension one): An edge of the polyhedron is divided into two new edges by taking an inner point of the edge as an additional vertex.
- CO1 (Composition of dimension one): Inverse of SU1.
- SU2 (Subdivision of dimension two): Two vertices of a polygon in the polyhedron are connected by a new edge dividing the polygon into two new polygons.
- CO2 (Composition of dimension two): Inverse of SU2.

Two polyhedra P and P' are said to be *elementarily related* if P can be transformed into P' by using a finite number of SU1, CO1, SU2, and CO2. A polyhedron is *orientable* if one can choose an orientation for each polygon such that each edge is used in both possible directions. The alternating sum $E(P) = \alpha_0 - \alpha_1 + \alpha_2$ for a polyhedron P is called its *Euler characteristic*. Here α_0, α_1, and α_2 denote the number of vertices, edges, and polygons, respectively. (In Σ, α_2 = the number of rows and α_1 = the number of different letters.)

By the fundamental theorem of the classification of 2-manifolds [24], each polyhedron is elementarily related to one of the following normal forms:

$$(H_0) \quad a\, a^{-1}$$

$$(H_p) \quad a_1\, b_1\, a_1^{-1}\, b_1^{-1}\, \ldots\, a_p\, b_p\, a_p^{-1}\, b_p^{-1}$$

$$(C_q) \quad c_1\, c_1\, c_2\, c_2\, \ldots\, c_q\, c_q$$

Here, H_0 is the normal form for a sphere (Euler characteristic = 2). H_p is the normal form for a p-tori, $p = 1, 2, \ldots$ (Euler characteristic = $2 - 2p$). Finally, C_q is the normal form for a nonorientable surface (Euler characteristic = $2 - q$, $q = 1, 2, \ldots$).

Simple topological shapes in $\top\flat$

Consider a rectangle and assign a definite sense of direction (orientation) to the perimeter. Denote the sides by a, b, a, b in that order. Place an arrow on each side such that the two sides a (or b) are made to coincide and such that the heads of the arrows coincide as well. The row $aba^{-1}b^{-1}$ is considered as a symbolic description of a polyhedron (e.g., a torus). It represents the four sides of the rectangle. (As we have noted earlier, the notation a^{-1} in above formula means that the third side is denoted by a but its arrow points opposite to the given orientation of the rectangle.)

As another example, consider a *lune*, i.e., a polygon with only two sides. Put arrows on the two sides such that the symbolic row reads aa^{-1}. This defines a division of the sphere into just one lune. That is, a sphere can be represented in topology as aa^{-1} and this is equivalent to a 2-gon (cf. Figure 1). A sphere can be represented in $\top\flat$ by the help of two curves (a contour c and a trajectory t).

A torus can be denoted $aba^{-1}b^{-1}$ as a symbolic representation and with a rectangle as a topological representation. Figure 2 shows the representation of the torus as a rectangle and prescribes how it can be constructed by the help of the rectangle. Figure 3 shows the representation of the torus by the help of the contour and the trajectory curves. Figure 4 depicts the representations of 2-tori in topology and $\top\flat$. Symbolic representation of a 2-tori is $aba^{-1}b^{-1}cdc^{-1}d^{-1}$. Of course, it is difficult to obtain an n-tori in this way. In the sequel, we'll show how an n-tori (a.k.a. a sphere with n handles) is obtained by the help of its symbolic representation.

There are also polyhedra which cannot be embedded in 3-space without penetrating themselves. Consider a rectangle with the symbolic row $baba^{-1}$ (Figure 5a). If we first make the identification along the sides labeled b, we get a *Möbius strip* [24]. Figure 5b shows the curve representation of a Möbius strip in $\top\flat$. The contour c is twisted by the twist curve tw while it is swept along the trajectory t.

If we first identify the sides labeled a in $aba^{-1}b$ (Figure 6a), we obtain a tube for which the two end circles are denoted by b. But to identify the two end circles, it is necessary to match the arrows on them. Deform the tube so that one of the ends is somewhat smaller than the other and penetrate this smaller end through the wall of the tube. Then bend the larger end a little toward the interior and the smaller end a little toward the exterior, and attach them according to the required mode of identification. We are left with what is commonly known as a *Klein bottle* [24]. Figure 6b shows the curve representation of a Klein bottle in $\top\flat$. The contour c is scaled by the help of the profile curve pr while it is swept along the trajectory t.

THE SWEEP PARADIGM

A class of solids called *nonprofiled sweep objects* is defined by two parametric curves: a 2-D *contour* (cross-section) and a 3-D *trajectory*. The contour is moved along the trajectory to generate a parametric surface [19]. A classification of sweep objects is given by Choi and Lee [13]. Other mathematical models of sweep objects can be found in works by by Post and Klok [23], Martin and Stepson [21], and Wang and Wang [25]. A fine classification is offered by Bronsvoort, Nieuwenhuizen, and Post [12]:

- Translational sweep: The contour is arbitrary and the trajectory is a straight line.
- Rotational sweep: The contour is arbitrary and the trajectory is a circle.
- Circle sweep: The contour is circular and the trajectory is arbitrary.
- General sweep (generalised cylinder): Both the contour and the trajectory are arbitrary. This can be further classified as (i) nonprofiled general sweep, (ii) profiled general sweep, (iii) depth-modulated general sweep, and (iv) twisted-profiled general sweep.

Nonprofiled general sweep

Nonprofiled general sweep is defined by an arbitrary closed 2-D contour and an arbitrary 3-D trajectory [12, 23]. Here, the 2-D contour $\mathbf{c}$ can be defined in parametric form:

$$\mathbf{c}(v) = (\mathbf{c}_x(v), \mathbf{c}_y(v)) \quad v_1 \leq v \leq v_f \text{and} \mathbf{c}(v_1) = \mathbf{c}(v_f)$$

As a parameter, v varies from v_1 to v_f. The parametric functions $\mathbf{c}_x$ and $\mathbf{c}_y$ trace out the contour. The 3-D trajectory $\mathbf{t}$ can be defined in the parametric form:

$$\mathbf{t}(u) = (\mathbf{t}_x(u), \mathbf{t}_y(u), \mathbf{t}_z(u)) \quad u_1 \leq u \leq u_l$$

As a parameter, u varies from u_1 to u_l. The parametric functions $\mathbf{t}_x$, $\mathbf{t}_y$, and $\mathbf{t}_z$ trace out the trajectory. For nonprofiled general sweep, the contour $\mathbf{c}$ moves along the trajectory $\mathbf{t}$. ($\mathbf{c}$ remains constant along $\mathbf{t}$.)

The spatial relation between the contour and the trajectory must be defined at every point of the trajectory. Figure 7 illustrates a nonprofiled general sweep object. The contour plane is perpendicular to $\mathbf{e}_1$, the unit vector tangent to the trajectory. As $\mathbf{e}_3$, a fixed unit normal to the plane of the trajectory is chosen. (Note that $\mathbf{e}_2$ is the vector product of $\mathbf{e}_3$ and $\mathbf{e}_1$.) A profiled general sweep object is given in Figure 8.

It is possible to present a nonprofiled general sweep as a vector function $\mathbf{T}_{np}$ of two parameters u and v:

$$\mathbf{T}_{np}(u,v) = \mathbf{Sweep}(\mathbf{t}(u), \mathbf{c}(v)) \quad u_1 \le u \le u_l \quad v_1 \le v \le v_f$$

For the sweep operation, it is necessary to use translation and 3-D rotation. The direction of the tangent vectors for the first and the last points of the trajectory must be selected as directions of the first and the last segments to rotate the contour. For other points, the direction of the vectors as specified by the previous and the next points must be selected.

Profiled general sweep

If deformations are necessary on a nonprofiled general sweep by scaling, then a profiled general sweep can be defined. The contour can be scaled independently in x and y directions. For each point of the trajectory, two scale factors $\mathbf{S}_x$ and $\mathbf{S}_y$ are specified. The scale functions are expressed in terms of the parameter of the trajectory:

$$\mathbf{S}(u) = (\mathbf{S}_x(u), \mathbf{S}_y(u)) \quad u_1 \le u \le u_l$$

If the scale factor is substituted in $\mathbf{T}_{np}$, it will have an effect only on the contour functions and the resultant profiled general sweep will be defined as a vector function $\mathbf{T}_p$:

$$\mathbf{T}_p(u,v) = (\mathbf{Sweep}(\mathbf{t}(u), \mathbf{c}(v)), \mathbf{S}(u)) \quad u_1 \le u \le u_l \quad v_1 \le v \le v_f$$

Depth-modulated general sweep

Sometimes it is necessary to give a third dimension to the sweep objects. In this case, a depth-modulated general sweep can be given as a vector function $\mathbf{T}_d$:

$$\mathbf{T}_d(u,v) = (\mathbf{Sweep}(\mathbf{t}(u), \mathbf{c}(v)), \mathbf{D}(\mathbf{t}_z(u))) \quad u_1 \le u \le u_l \quad v_1 \le v \le v_f$$

Twisted-profiled general sweep

If the contour curve is deformed by twisting and scaling while sweeping along the trajectory curve, the produced surface is called a twisted-profiled general sweep object. The contour can be twisted independently in x and y directions for each point of the trajectory. For this purpose, two twist factors $\mathbf{T}_{wx}$ and $\mathbf{T}_{wy}$ can be specified. The twist functions are expressed in terms of the parameter of the trajectory:

$$\mathbf{T}_w(u) = (\mathbf{T}_{wx}(u), \mathbf{T}_{wy}(u)) \quad u_1 \le u \le u_l$$

If the twist factor is substituted in $\mathbf{T}_p$ defined earlier, it will have an effect only on the contour functions and the resultant twisted-profiled general sweep will be defined as a vector function $\mathbf{T}_{wp}$:

$$\mathbf{T}_{wp}(u,v) = (\mathbf{Sweep}(\mathbf{t}(u), \mathbf{c}(v)), \mathbf{T}_w(u), \mathbf{S}(u))) \quad u_1 \le u \le u_l \quad v_1 \le v \le v_f$$

We now cite some works on deformation which influenced our work. Coquillart presents an offset technique for control points of rational B-spline curves to produce (profiled) sweep objects [14]. Choi and Lee use simple transformations and blending to represent more complex sweep objects [13]. Woodward presents a skinning technique to produce 3-D shapes that resemble blended sweep objects [26].

A discrete model for sweeping

A sweep surface $Sw(C,T)$ is produced by moving a given contour curve C along a given trajectory curve T. The plane of C must be perpendicular to T at any point of T. C may be any planar closed or open curve and T may be any 3-D closed or open curve. If C is a closed curve and T is a 3-D curve segment, then the produced sweep object is called a *generalised cylinder* [14].

If C is deformed by scaling with a scale factor as it is swept along T, then the produced sweep object is called a *profiled general sweep* object. Here, we accept as the scale factor the distance of a given curve P to any selected axis. So, a profiled general sweep surface can be defined as $Sw_p(C,T,P)$. Here, P represents a profile curve which has the same number of points with T.

If C is deformed by twisting and scaling while sweeping along T, the produced object is called a *twisted-profiled general sweep object*. We define the twist factor by a curve Tw. This is a planar curve which has the same number of points with T and is mapped between the minimum and the maximum values of the twisting angles only at one dimension. Then, a twisted-profiled general sweep surface can be defined by four curves, viz. $Sw_{tp}(C,T,P,Tw)$.

In the following, we present a discrete mathematical definition of twisted-profiled general sweeping. In this definition, we use the conventional matrix notation to represent the transformations and obtain a general sweep matrix as a result. In our representation, the matrix twists, scales, and sweeps a contour curve C along a trajectory curve T to handle a grid of general sweep surface points. We'll formulate the discrete mathematical definitions of curves (C,T,P,Tw) as vector functions.

C is a planar curve or polygon with n points and each point $(x_i, y_i, 0)$ of C can be represented as $C_i,\ i = 1,2,\ldots,n$. T is any 3-D curve or polygon with m points and (without loss of generality) the first point of T is in the origin. Each point (x_j, y_j, z_j) of T can be represented as $T_j,\ j = 1,2,\ldots,m$. P can be given as a 1-D position vector with m component vectors that include scaling factors. But here, we take P as a planar curve and later calculate the scaling factors by the help of the curve. This may be useful for interactive systems. Each point $(x_j, y_j, 0)$ of P can be represented as $P_j,\ j = 1,2,\ldots,m$. Tw can be also given as a 1-D position vector with m vectors that include twisting factors. We choose to specify Tw as a planar curve and later calculate the twisting factors by the help of the curve. Each point $(x_j, y_j, 0)$ of Tw can be represented as $Tw_j,\ j = 1,2,\ldots,m$. In the sequel, we give a procedure to handle a twisted-profiled general sweep object. In this procedure, C is a curve centered at the origin and the first point of T is also at the origin. In other cases, they must be translated to origin and then the generated curves must be translated back to their actual positions at the end of the procedure. First, C is rotated around the z-axis by Tw for twisting. Second, C is scaled by P. Third, C is rotated around the x, y, and z-axes by angles found for each point of T. Finally, each twisted, scaled, and rotated C is translated to every point of T.

Twisting

We have taken the twisting factor Tw as a planar curve. But, Tw represents only the twisting angles for each point of the trajectory curve T. That is, it can be represented as a 1-D vector function linearly interpolated at only one dimension of Tw. Tw can be drawn interactively as a planar curve in an interactive system and then is interpolated at one dimension to handle twisting angles. The interpolated 1-D vector function θ includes rotating angles for each point of T and can be shown as θ_j, $j = 1, 2, \ldots, m$. The following matrix rotates a contour curve C that is centered at the origin (for twisting as a function of θ):

$$\begin{bmatrix} R_{tw} \end{bmatrix} = \begin{bmatrix} c\theta_j & s\theta_j & 0 & 0 \\ -s\theta_j & c\theta_j & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \begin{array}{l} c\theta_j \overset{Df}{=} \cos\theta_j \\ s\theta_j \overset{Df}{=} \sin\theta_j \end{array}$$

Scaling

We have taken a profile curve P (a planar curve) as the scaling factor. P also represents only one magnitude for each point of the trajectory curve T. Then, it can also be written as a 1-D vector function which takes the distances of the vertices of P to the vertical (or horizontal) axis at only one dimension. For an interactive system, first, an axis is given for reference and P is drawn interactively. Then, distances of vertices of P at one dimension to the axis give the scaling factor for C. Scaling factor s is a 1-D vector function and includes scaling radii for points of T. It can be shown as s_j, $j = 1, 2, \ldots, m$. The following matrix scales a contour curve C that is centered at the origin, as a function of s:

$$\begin{bmatrix} S_p \end{bmatrix} = \begin{bmatrix} s_j & 0 & 0 & 0 \\ 0 & s_j & 0 & 0 \\ 0 & 0 & s_j & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Rotation

The twisted-profiled contour curve C must be rotated in 3-D with respect to the tangent vector at each point of trajectory curve T. That is, the plane of C must be made perpendicular to the tangent vector of T at each point. We calculate the discrete derivative for each point of T to handle the tangent vector. To this end, we add two dummy vertices T_0 and T_{m+1} to T. ($T_0 = T_1$ and $T_{m+1} = T_m$.) In the following, we use matrices R_x, R_y, and R_z to handle the real position of C and Tx, Ty, and Tz to represent the x, y, and z distances of the vertices of T.

Rotation around the x-axis:

$$\begin{bmatrix} R_x \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c\alpha_j & s\alpha_j & 0 \\ 0 & -s\alpha_j & c\alpha_j & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \begin{array}{l} c\alpha_j \overset{Df}{=} \cos\alpha_j = \frac{Ty_{j-1} - Ty_{j+1}}{h_x} \\ s\alpha_j \overset{Df}{=} \sin\alpha_j = \frac{Tz_{j+1} - Tz_{j-1}}{h_x} \end{array}$$

$$h_x = \sqrt{(Ty_{j-1} - Ty_{j+1})^2 + (Tz_{j-1} - Tz_{j+1})^2}$$

Rotation around the y-axis:

$$\left[\ R_y\ \right] = \begin{bmatrix} c\beta_j & 0 & -s\beta_j & 0 \\ 0 & 1 & 0 & 0 \\ s\beta_j & 0 & c\beta_j & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \begin{array}{l} c\beta_j \overset{Df}{=} \cos\beta_j = \frac{Tx_{j-1}-Tx_{j+1}}{h_y} \\ s\beta_j \overset{Df}{=} \sin\beta_j = \frac{Tz_{j+1}-Tz_{j-1}}{h_y} \end{array}$$

$$h_y = \sqrt{(Tx_{j-1} - Tx_{j+1})^2 + (Tz_{j-1} - Tz_{j+1})^2}$$

Rotation around the z-axis:

$$\left[\ R_z\ \right] = \begin{bmatrix} c\gamma_j & s\gamma_j & 0 & 0 \\ -s\gamma_j & c\gamma_j & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \begin{array}{l} c\gamma_j \overset{Df}{=} \cos\gamma_j = \frac{Tx_{j-1}-Tx_{j+1}}{h_z} \\ s\gamma_j \overset{Df}{=} \sin\gamma_j = \frac{Ty_{j+1}-Ty_{j-1}}{h_z} \end{array}$$

$$h_z = \sqrt{(Tx_{j-1} - Tx_{j+1})^2 + (Ty_{j-1} - Ty_{j+1})^2}$$

Translation

Finally, C must be translated to each point of T to finish the sweeping operation. We use a 3-D translation matrix as follows:

$$\left[\ T_{xyz}\ \right] = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ a & b & c & 1 \end{bmatrix} \qquad \begin{array}{l} a \overset{Df}{=} Tx_j - Tx_1 \\ b \overset{Df}{=} Ty_j - Ty_1 \\ c \overset{Df}{=} Tz_j - Tz_1 \end{array}$$

The mathematical model

A general twisted-profiled sweeping matrix Sw_{tp} is obtained, if the above transformation matrices are multiplied in the following order:

$$\left[\ Sw_{tp}\ \right] = \left[\ R_{tw}\ \right]\left[\ S_p\ \right]\left[\ R_x\ \right]\left[\ R_y\ \right]\left[\ R_z\ \right]\left[\ T_{xyz}\ \right]$$

After some manipulations, the following matrix is computed:

$$\left[\ Sw_{tp}\ \right] = \begin{bmatrix} \begin{array}{c} s_jc\theta_jc\beta_jc\gamma_j \\ +s_js\theta_js\alpha_js\beta_jc\gamma_j \\ -s_js\theta_jc\alpha_js\gamma_j \end{array} & \begin{array}{c} s_jc\theta_jc\beta_js\gamma_j \\ +s_js\theta_js\alpha_js\beta_js\gamma_j \\ +s_js\theta_jc\alpha_jc\gamma_j \end{array} & \begin{array}{c} -s_jc\theta_js\beta_j \\ +s_js\theta_js\alpha_jc\beta_j \end{array} & 0 \\ \begin{array}{c} -s_js\theta_jc\beta_jc\gamma_j \\ +s_jc\theta_js\alpha_js\beta_jc\gamma_j \\ -s_jc\theta_jc\alpha_js\gamma_j \end{array} & \begin{array}{c} -s_js\theta_jc\beta_js\gamma_j \\ +s_jc\theta_js\alpha_js\beta_js\gamma_j \\ +s_jc\theta_jc\alpha_jc\gamma_j \end{array} & \begin{array}{c} s_js\theta_js\beta_j \\ +s_jc\theta_js\alpha_jc\beta_j \end{array} & 0 \\ \begin{array}{c} s_jc\alpha_js\beta_jc\gamma_j \\ +s_js\alpha_js\gamma_j \end{array} & \begin{array}{c} s_jc\alpha_js\beta_js\gamma_j \\ -s_js\alpha_jc\gamma_j \end{array} & s_jc\alpha_jc\beta_j & 0 \\ a & b & c & 1 \end{bmatrix}$$

Consequently, a twisted-profiled sweep surface can be calculated in the following form. First, each point of C must be evaluated for matrix Sw_{tp}, and then, a new matrix Sw_{tp} must be modified for each incremented j.

$$\left[\ Sweep\ \ surface\ \right]_{i,j,\theta,s} = \left[\ C\ \right]_i \left[\ Sw_{tp}\ \right]_{j,\theta,s} \qquad 1 \le i \le n \quad 1 \le j \le m$$

In the above formula, j varies slowly, i.e., we compute the formula for a particular j and every i.

For the planar case, if C is a 2-D curve or a polygon centered at the origin in the x-y plane and T is a curve or polygon starting at the origin in the x-z plane, then a planar twisted-profiled sweep surface can be computed with the help of following matrix:

$$\left[\ Sw_{ptp}\ \right] = \left[\ R_{tw}\ \right]\left[\ S_p\ \right]\left[\ R_y\ \right]\left[\ T_{xz}\ \right]$$

After some manipulations, the following is obtained:

$$\left[\ Sw_{ptp}\ \right] = \begin{bmatrix} s_j c\theta_j c\beta_j & s_j s\theta_j & -s_j c\theta_j s\beta_j & 0 \\ -s_j s\theta_j c\beta_j & s_j c\theta_j & s_j s\theta_j s\beta_j & 0 \\ s_j s\beta_j & 0 & s_j c\beta_j & 0 \\ a & 0 & b & 1 \end{bmatrix}$$

Sweeping with varying contour

A twisted-profiled sweep surface can be calculated in the following form. First, each point of C must be evaluated for matrix Sw_{tp}, and then, a new matrix Sw_{tp} must be modified for each incremented j:

$$\left[\ Sweep\, surface\ \right]_{i,j,\theta,s} = \left[\ C\ \right]_i \left[\ Sw_{tp}\ \right]_{j,\theta,s} \qquad 1 \le i \le n \quad 1 \le j \le m$$

In the above formula, j varies slowly, i.e., we compute the formula for a particular j and every i. The contour curve C is a constant curve along the trajectory curve T. But frequently C must be a different curve for some points of T to handle more complex sweep objects. Our approach to resolve this situation is as follows.

First, different contour curves are defined for some points of T. Second, these curves are linearly interpolated respectively to obtain new contour curves for each point of T. Third, the obtained 2-D discrete curves are used for the sweep operation. (That is, a different contour curve is used for each point of T to obtain sweep surfaces with varying contour.)

A twisted-profiled sweep surface with varying contour can be calculated in the following form:

$$\begin{bmatrix} Sweep\ \ surface \\ with\ \ varying\ \ contour \end{bmatrix}_{i,k,j,\theta,s} = \left[\ C\ \right]_{i,k} \left[\ Sw_{tp}\ \right]_{j,\theta,s} \qquad \begin{matrix} 1 \le i \le n \\ 1 \le j,k \le m \end{matrix}$$

In the above formula, a different contour curve is taken into account for each point of the trajectory curve.

SOFTWARE CHARACTERISTICS OF T♭

In this section, the user interface system of T♭ and some ideas on which the system is based will be presented. A user interface management system is a tool set designed to encourage cooperation in the rapid development, tailoring, and management of the interaction in an application domain across varying devices, interaction techniques, and user interface styles. There are elements that are presented to the programmer by user interface systems and play an effective role to provide user computer interaction. In general, these *interactors* include windows, active or passive text areas, pulldown menus, buttons, mouse, light pen, and other pointing devices, scrollbars, etc. A sample user interface, or part of a complex user interface, can be realised by a small, fixed collection of interactors. Such a collection is called a *dialog*.

T♭ has a rather straightforward user interface. Figure 9 shows the general appearance of this interface. Since T♭ is a research tool, speed is preferred over comprehensibility and orthogonality. T♭'s user interface is based on the following guidelines:

- Interactors with a major functionality are shown on the screen all the time. Interactors with similar purposes were collected together. They were designed as buttons and placed in the top side of the main window. They are continuously shown on the screen so that the user immediately can activate any of them.
- Excluding text operations, only the mouse is used for interactions.
- It is made sure that the interactors do not occupy too much space on the screen.
- Interactors and their positions are good-looking and well-placed. Different colours, shapes, and fonts have been used for interactors to decrease the accommodation time of the user.
- It is made sure that the interactors have not too many items.
- To increase the speed of the system, sometimes interactors with a special purpose are defined.
- There are two types of help menus used in T♭. In the first type, help menus are resident and tell the user what they will do. In the second, each interactor has its own help text which flashes or beeps to warn the user.

Available functionalities

Our fundamental aim is to hide the details of the processing from the user. This is in the precise spirit of Pentland's sketching system [22]. The main goal of our work is to provide a rudimentary graphical workbench to help topologists, artists, solid modelers, and geographers to illustrate their ideas more effectively.

In this section, the power of T♭ is presented. A general appearance of main interactors of T♭ is shown in Figure 9. Each interactor has many subinteractors.

Grid selection

A general sweep object is represented by two curves (contour and trajectory) in T♭. Contour and trajectory curves usually have many discrete points. The number of points on the contour and the trajectory curves determines the number of grids of the produced sweep object. So, the number of discrete points in auxiliary curves must be determined by the user. Menu GRID lets the user select this number.

The profiled, depth-modulated, and twisted sweep objects have a profile, a depth modulation, and a twist curve, respectively. The number of points on these curves is set automatically to the number of points of the trajectory curve. The help menu gives information about how the selection is done and how many points there are on the contour and the trajectory curves.

Drawing

Curves are specified and drawn interactively by the help of a mouse. They can be produced in two ways: free form and approximated form. In the former, a curve is produced by entering points of the curve by the mouse. There is no further operation to smooth the curve. In the latter, a curve is produced by entering each control point of the curve by the mouse. In this case, the curve is approximated with the use of Bézier or B-spline algorithms (and the produced curve is smooth).

The following sweep objects can be created by the help of drawing menu:

- Rotational sweep objects
- Nonprofiled sweep objects
- Profiled sweep objects
- Depth-modulated sweep objects
- Twisted-profiled sweep objects
- Assorted knots

In the following, we only give directions for creating a nonprofiled sweep object in detail by using the button DRAWING. Each of the other objects can be created via similar selections.

Inputs to produce a nonprofiled sweep object are two curves. The output is a 3-D object obtained by moving or sweeping one curve along the other (Figure 10). The method includes both translational and rotational sweep and provides the user with an opportunity to create complex 3-D objects. Curves that are sequences of points in space are stored in a 1-D array in T♭. They can also be determined in any other way such as a B-spline method or by inputting a sequence of 2-D points. One curve is considered to be a path on which the other curve moves while generating a surface. The main idea of the method involves two transformations, namely a 3-D translation and a 3-D rotation. The first curve is replicated on the second curve by translating the contour to the points of the trajectory as many times as the number of points given for the trajectory. In other words, one copy of the contour curve is generated for each point of the trajectory curve and is moved to these points. Next, the replicated curves are rotated by the position of the point.

The user must select the button DRAWING to produce a sweep object. After the selection of the button, a menu is displayed. User must select the button NONPROFILED SWEEP to create a nonprofiled sweep object. Some submenus such as FREE_FREE, FREE_BEZIER, BEZIER_FREE, and BEZIER_BEZIER are shown on the screen. The button BEZIER_FREE inform that the contour curve will be drawn by using the Bézier algorithm and the trajectory curve will be drawn as a free form curve. After the selection of any menu, the contour and the trajectory curves of the nonprofiled sweep object is drawn. As a result, T♭ uses curves to create a nonprofiled sweep object.

The second interactive method to produce swept shapes is well-known and commonly called *simple rotational sweep.* Let p be a point in 3-D. If p is rotated about an axis ℓ the resultant figure is a circle in 3-D. Similarly, if one takes a line segment s and rotates it about ℓ a cylindrical object is obtained. Now, take a circle c and rotate it about ℓ. The resultant figure is a torus. Mouse-picked control points are used to define an arbitrary curve segment which in turn is rotated about ℓ. Many interesting surfaces of revolution can be computed in this way (Figure 11).

Although the above methods are restricted to 2-D curves, they can be extended to the 3-D case. For this, we give a third method called *nonplanar general sweep.* An important aspect of the method is to enter the 3-D points by the help of a mouse. This is provided by a third curve, appropriately called a *depth-modulation curve.* The curve gives a nonplanarity to the trajectory and now, the second curve is exactly a 3-D curve (Figure 12). Some knots, spirals, and other nonplanar complicated objects can be produced by the help of this method.

The fourth method is *profiled general sweep.* Three curves are necessary for this method: a contour, a trajectory, and a profile curve. The contour curve is scaled to different sizes according to the profile curve while sweeping is performed (Figure 13).

The last method is a *twisted-profiled nonplanar general sweep.* Five curves are necessary for this method: a contour, a trajectory, a depth-modulation curve, a profile, and a twisting curve. The contour curve is twisted and scaled to different angles and sizes according to the profile curve, while sweeping is performed (Figure 14).

Twisting

Essentially the menu TWIST is also an editing menu. A user can edit the twist curve of a sweep object or create a new twist curve by selecting the twist button. To this end, first an object is selected by the help of the mouse in the current scene. Then, contour and trajectory (and profile or depth-modulation curves, if necessary) curves and maximum and minimum boundaries of the twist curve are shown on the screen. After this, a circular twist submenu is shown on the twist menu.

Skinning

At least three curves are necessary for this method. These are the contour curve, the trajectory curve and another contour curve. The number of contour curves is defined by user. That is, the user can draw many contour curves to determine the shape of the produced object. Two contour curves are linearly interpolated to obtain new contour curves for each point of the trajectory curve and then sweeping is performed by using these curves. If many contour curves are drawn, each curve is linearly interpolated to obtain new contour curves. The curves are swept along the trajectory curve and a new sweep object with different contour is produced (Figure 15).

Transformations

$\top\flat$ can perform 3-D rotation, scaling, and translation to give the required position or shape to the produced sweep object. The menu XFORM presents rotation, scaling, and translation options to the user. Rotation can be performed in two ways. In the former, only one object is selected on the screen. After this, a rotation menu is displayed on the screen to select angles of rotation around x, y, or z-axis. After the selection of the rotation angle, the rotation is performed. In the latter, many objects can be selected to rotate by the same rotation angle. It is also possible to select two or three angles to rotate the object around the x, y, or z axes at the same time.

The button SCALE must be selected to scale an object. After selecting an object with the help of the mouse, a scrollbar is displayed on the screen. A number is selected on the scrollbar as the scaling coefficient. If the selected number is negative, then the selected object is made smaller. If the number is positive, the object is enlarged.

For the translation of an object, the button TRANSLATE must be selected. The selected object is translated to the required place with the help of the mouse.

Shading

The generation of a sweep object finishes as soon as the points of the surface are computed. Then the resultant surface is shaded by constant or Gouraud shading. Constant shading is fast and adds enough realism to the obtained image. Hidden surfaces are eliminated using the z-buffer method. Gouraud shading is slower than constant shading but provides a good enough effect toward realism for the generated surface.

Blending

If just one sweep operation is not sufficient to obtain a complicated object, two sweep objects can be blended to produce the required effect [17]. There are three interactors to perform different connections. Two different sweep objects can be blended by selecting one edge of each object if the user selects the first button. The user must draw a path between the selected edges. Only the control points of the curve must be given to obtain a smooth path. An approximated curve is obtained between the objects and sweeping is performed by using the curve and the edges of the objects. If the user selects the second button, the blending operation is performed between a curve selected on a sweep object and another curve drawn by the user. The user must also draw a path between the curves. A new object (with one edge on the selected object) is produced. Two curves with different shapes are blended if the user selects the third button. The user must draw two different curves and a trajectory between these curves.

Local deformations

An easy method was developed to perform local deformations in T♭. For this, the button LOCAL DEFS. must be selected. Then, the button BEZIER or the button FREE from the displayed submenus (Figure 16) must be selected to determine the type of auxiliary deformation curves. After that, the object that will be deformed must be selected using the mouse. Only the contour and the trajectory curves of the object are shown on the screen. The user must draw two auxiliary curves along the trajectory curves. Local deformation is performed using auxiliary curves. Figure 16 shows the contour c, the trajectory t, and deformation curves d_1 and d_2, and the locally deformed sweep object.

Handle normalisation

Users must select the button TOPOLOGY to normalise the handles in the produced symbols (which represent a polyhedron) and to see the normalised handles and graphical representation of these handles (i.e., a sphere with numerous handles) on the screen. The user must only give the number of topology symbols by using the topology menu (Figure 17). T♭ creates random symbols with the given number. Then, it computes the normal form of the symbols to obtain true handles. It prints the obtained handles and draws a sphere with these handles (Figure 17). T♭ uses the classical algorithm given in Ringel [24] (also see Abelson and diSessa [1], and Griffiths [16]) to compute the normal form of a given polyhedron.

Animation

There are two ways for animating sweep objects in T♭. In the former, only one operation is performed for animation. In this case, the object to be animated is produced by using the modeler of T♭. In the latter, both modeling and animation are performed at the same time. If necessary, each produced object can be shaded.

The idea of animation of a sweep object is based on the linear interpolation of curves. Two animation curves are necessary to animate a sweep object. Each point of the curves is linearly interpolated and many new curves are produced. The first point of the first curve must be linearly interpolated to the first point of the second curve, the second point of the first curve must be linearly interpolated to the second point of the second curve, and so on.

Figure 18 shows animation by the help of a profile curve (two profile curves are necessary). Curves are interpolated to produce new profile curves by an operation that is similar to the operation mentioned above. The contour curve is scaled by each produced profile curve while a sweep is performed. Figure 19 shows animation by the help of a depth-modulation curve. Two depth-modulation curves are interpolated and new depth-modulation curves are produced. The third dimension for each point of the trajectory curve is determined by each newly produced depth-modulation curve while sweeping is performed to animate a sweep object.

Figure 20 shows animation by the help of a twist curve. Twist curves are linearly interpolated and the contour curve is twisted by the help of each new twist curve.

SUMMARY

We have compared the curve representation of a surface in T♭ and its symbolic representation in topology. A new mathematical model was presented for the planar and the nonplanar twisted-profiled general sweep objects. In addition, the model was generalised for sweep objects with varying contours. Figure 21 shows some rotational sweep objects. Figures 22 and 23 show nonprofiled and profiled sweep objects, respectively. Figures 24 and 25 show some depth-modulated and twisted sweep objects, respectively.

Global or local deformations can be performed interactively on a produced sweep object. For example, a small ‘bump’ or a tiny ‘well’ can be created on a given surface, e.g., a torus. A locally or globally deformed object can be deformed repeatedly. Figure 26 shows some sweep objects with local deformations.

It is possible to produce sweep objects with a varying contour. Several contour curves with different shapes can be assigned interactively to the trajectory curve to produce the varying contour. Figure 27 shows some examples.

Transparent windows can be created on a sweep object to inspect the sublayers of it. We use a special data structure to open the windows on the object while shading is being performed. It is possible to create a rectangular (or a polygonal) window on the object. Figure 28 shows an object with transparent windows on it.

Sweep objects produced by T♭ can be combined to produce a more complicated object. This can be done in two ways. In the former, the base sweep objects are produced and the objects are transformed to provide the required connections with each other. (Figures 29, 30, and 31 show some connected objects produced in this way.) In the latter, the sweep objects are produced and blended or connected with each other to produce a more complicated object. Some figures produced in this way are as follows.

Figure 32 was digitised from Francis’ book (p. 150) and Figure 33 was produced by T♭. Figure 34 includes two pictures representing a knot. The first picture was digitised from Francis’ book (p. 38) and the second was produced by T♭. Figure 35 (swapping handle cores) was digitised from Francis’ book (p. 140) and Figure 36 was produced by T♭.

Figure 37 shows two pictures (eight knot). The first one was digitised from Francis’ book (p. 37) and the second was produced by T♭.

As for future work, some higher level blending operations may be developed to obtain more complicated objects; blending in $\top\flat$ is naive and insufficient in its present form. Some other techniques (for example, B-spline surfaces with interactive operations, finite-element method, etc.) must be included to model complicated objects. Finally, a ray tracing option must be included in order to give a more realistic appearance to the produced objects [11].

ACKNOWLEDGEMENTS

The initial thinking of the second author on a topological picturebook has started in the Department of Interactive Systems, Centre for Mathematics and Computer Science (CWI), Amsterdam. He is grateful to Paul J. W. ten Hagen, Head of the Department, for his invaluable support in this regard.

This paper is a thoroughly revised version of the portions of the first author's doctoral thesis [5]. The reader is also referred to references [4, 6, 7] for preliminary versions which are significantly shorter.

References

[1] **Abelson, H and diSessa, A** *Turtle Geometry: The Computer as a Medium for Exploring Mathematics* MIT Press (1982)

[2] **Akman, V** *Geometry and Graphics Applied to Robotics* Technical Report No CS–R8727, Centre for Mathematics and Computer Science, Amsterdam (1987) [Also in **Earnshaw, R A (Ed.)** *Theoretical Foundations of Computer Graphics and CAD* NATO ASI Series Vol F40, Springer-Verlag, pp 619–638 (1988)]

[3] **Akman, V** *Steps into a Geometer's Workbench* Technical Report No CS–R8726, Centre for Mathematics and Computer Science, Amsterdam (1987) [Also in **van der Burgh, A H P and Mattheij, R M M (Eds.)** *Proceedings of First International Conference on Industrial and Applied Mathematics (ICIAM'87)* CWI Tracts Vol 36, Centre for Mathematics and Computer Science, Amsterdam, pp 257–275 (1987)]

[4] **Akman, V, Arslan, A and Franklin, W R** 'Implementing a topological picturebook' In *Proceedings of 13th IMACS World Congress on Computation and Applied Mathematics* Dublin (1991)

[5] **Arslan, A** *An Electronic Topological Picturebook* PhD dissertation, Department of Computer Engineering and Information Science, Bilkent University, Ankara (1992)

[6] **Arslan, A and Akman, V** 'Sweeping with all graphical ingredients' In **Baray, M and Özgüç, B (Eds.)** *Proceedings of Sixth International Symposium on Computer and Information Sciences (ISCIS VI)* Vol 2, Elsevier, pp 1225–1234 (1991)

[7] **Arslan, A and Akman, V** 'An electronic topological picturebook' In *Proceedings of NATO Advanced Study Institute on Cognitive and Linguistic Aspects of Geographic Space* Las Navas del Marques, Spain (1990)

[8] **Barsky, B A** *Computer Graphics and Geometric Modelling Using Beta-Splines* Springer-Verlag (1987)

[9] **Bartels, R H, Beatty, J C and Barsky, B A** *An Introduction to Splines for Use in Computer Graphics and Geometric Modelling* Morgan Kaufmann (1988)

[10] **Baumgart, B G** *GEOMED: Geometric Editor* Technical Report No STAN–CS–74–414, Computer Science Department, Stanford University, Stanford, CA (1974)

[11] **Bronsvoort, W F and Klok, F** 'Ray tracing generalised cylinders' *ACM Transaction on Graphics* Vol 4 No 4 pp 291–303 (1985)

[12] **Bronsvoort, W F, Nieuwenhuizen, P R V and Post, F H** 'Display of profiled sweep objects' *Visual Computer* Vol 5 No 3 pp 147–157 (1989)

[13] **Choi, B K and Lee, C S** 'Sweep surfaces modelling via coordinate transformations and blending' *Computer-Aided Design* Vol 22 No 2 pp 87–96 (1990)

[14] **Coquillart, S** 'A control-point-based sweeping technique' *IEEE Computer Graphics and Applications* Vol 7 No 10 pp 36–45 (1987)

[15] **Francis, G K** *A Topological Picturebook* Springer-Verlag (1987)

[16] **Griffiths, H B** *Surfaces* Cambridge University Press (1981)

[17] **Hartman, E** 'Blending of implicit surfaces with functional splines' *Computer-Aided Design* Vol 22 No 8 pp 500–507 (1990)

[18] **Hoffmann, C and Hopcroft, J** 'Automatic surface generation in computer-aided design' *Visual Computer* Vol 1 No 2 pp 92–100 (1985)

[19] **Klok, F** 'Two moving coordinate frames for sweeping along a 3-D trajectory' *Computer Aided Geometric Design* Vol 3 pp 217–229 (1986)

[20] **Kneale, D** 'Shaping ideas: A topologist wows the world of math by seeing the unseen' *Wall Street Journal* (March 18, 1983)

[21] **Martin, R R and Stepson, P C** 'Sweeping of three-dimensional objects' *Computer-Aided Design* Vol 22 No 4 pp 223–234 (1990)

[22] **Pentland, A P** *Supersketch User Manual* Artificial Intelligence Centre, SRI International, Menlo Park, CA (1985)

[23] **Post, F H and Klok, F** 'Deformations of sweep objects in solid modelling' In **Requicha, A A G (Ed.)** *Eurographics'86 Proceedings* North-Holland, pp 103–114 (1986)

[24] **Ringel, G** *Map Color Theorem* Springer-Verlag (1974)

[25] **Wang, W P and Wang, K K** 'Geometric modelling for swept volume of moving solids' *IEEE Computer Graphics and Applications* Vol 6 No 12 pp 8–17 (1986)

[26] **Woodward, C D** 'Skinning techniques for interactive B-spline surface interpolation' *Computer-Aided Design* Vol 20 No 8 pp 441–451 (1988)

**************** now come the figures *******************

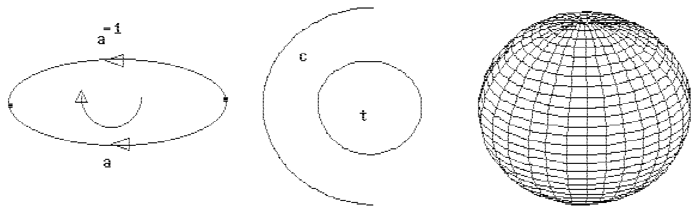


Figure 1 The representation of a sphere as a 2-gon aa^{-1} in topology. A sphere can be represented with two curves (a contour c and a trajectory t) in T♭.

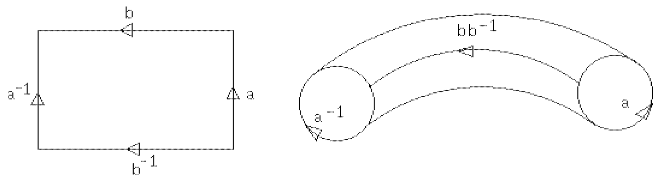


Figure 2 A rectangle represented by $aba^{-1}b^{-1}$ is transformed into a torus via an appropriate identification of the sides.

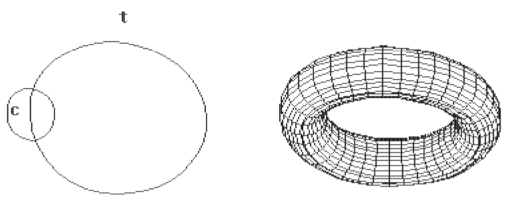


Figure 3 A torus can be defined by a contour curve c and a trajectory curve t in ⊤♭.

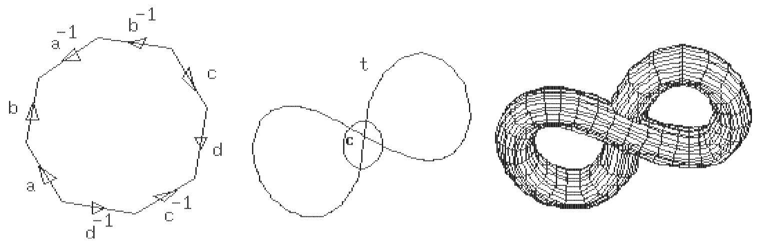


Figure 4 A 2-tori is represented as $aba^{-1}b^{-1}cdc^{-1}d^{-1}$ and can be defined by the appropriate c and t curves in ⊤♭.

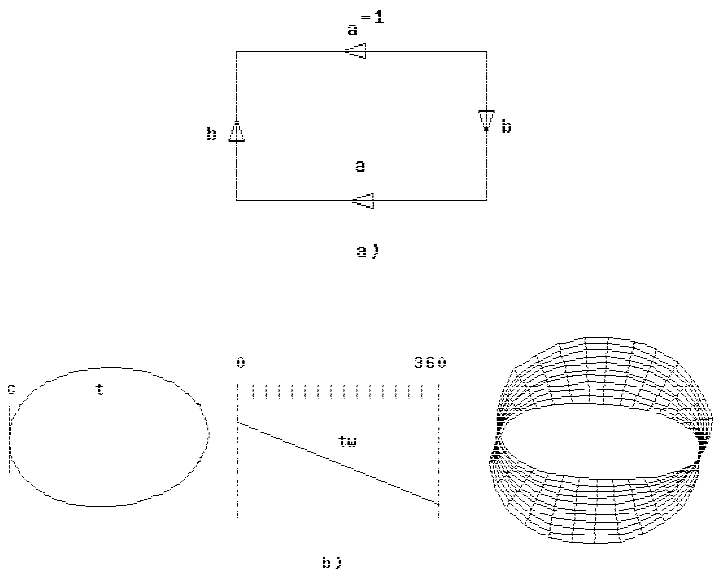


Figure 5 A Möbius strip is defined by a rectangle with $baba^{-1}$ and by a contour c, a trajectory t, and a twist curve tw in $\top\flat$.

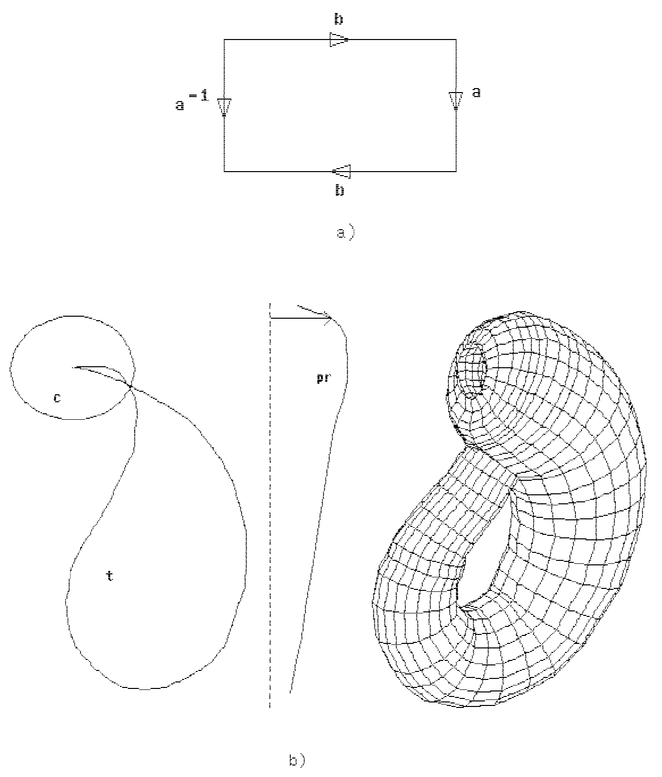


Figure 6 A Klein bottle is defined by a rectangle with $aba^{-1}b$ and by a contour c, a trajectory t, and a profile curve pr in T♭.

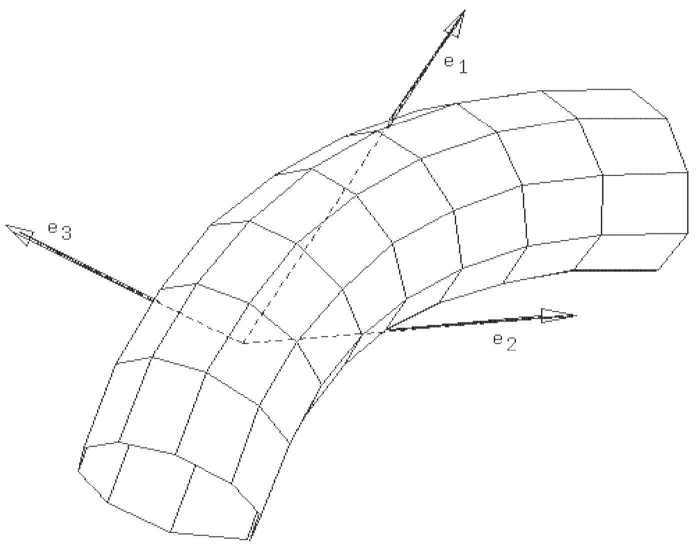


Figure 7 A nonprofiled general sweep object.

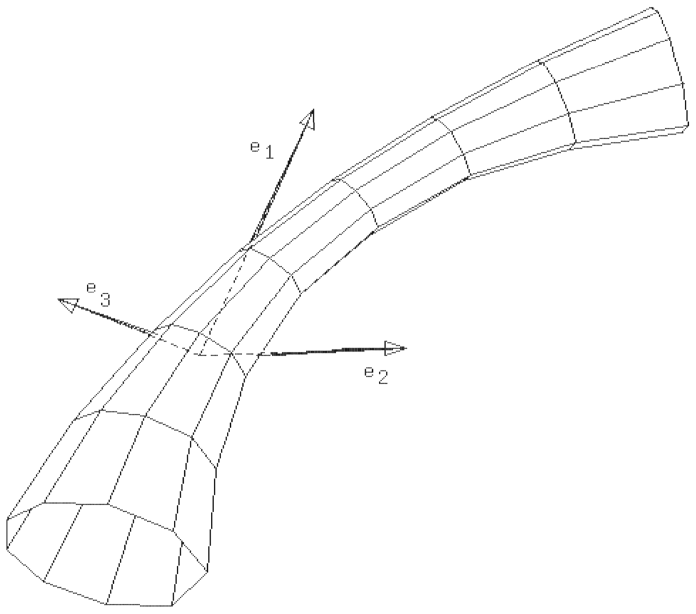


Figure 8 A profiled general sweep object.

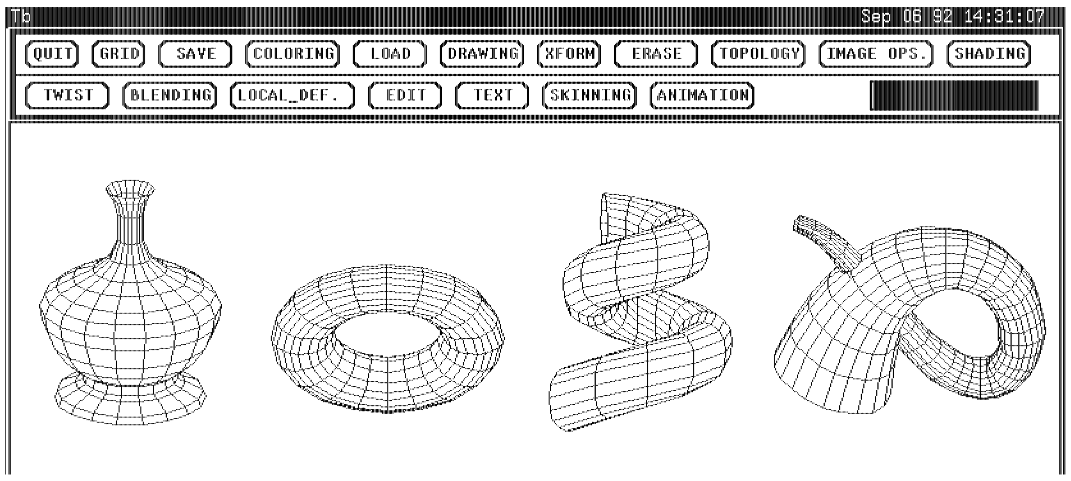


Figure 9 General appearance of the user interface system of T♭ and some 3-D sweep objects produced by T♭.

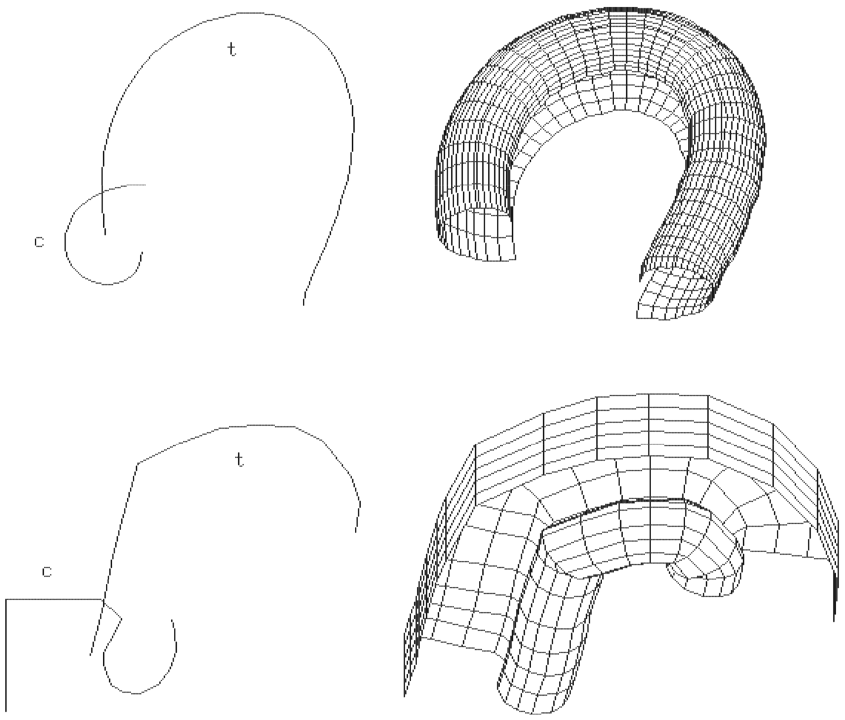


Figure 10 A contour is swept along a planar trajectory in order to generate a surface of desired shape. Both curves are defined arbitrarily.

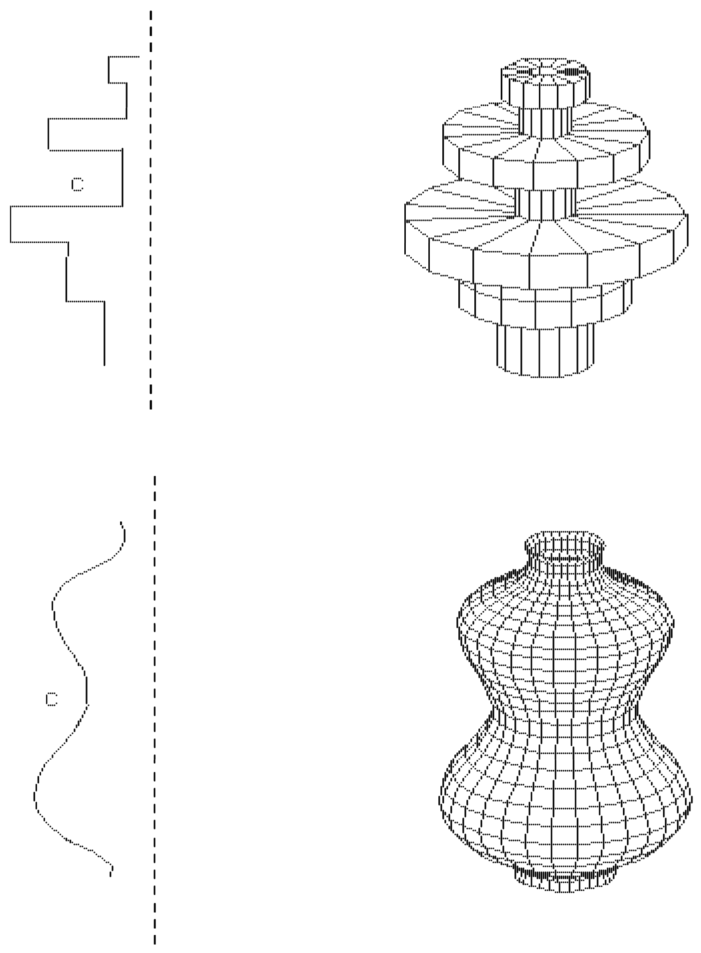


Figure 11 Both of the above objects are obtained by rotating a curve about an axis ℓ (shown in dotted lines). The first curve is a sequence of points, and the second one is a Bézier curve for which a set of control points is given.

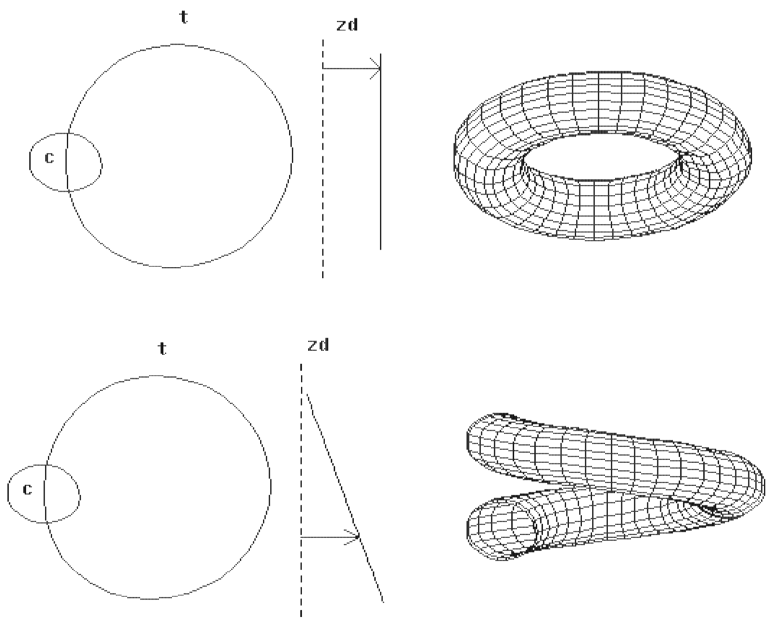


Figure 12 Contour *c*, trajectory *t*, and depth-modulation *zd* curves and the produced objects. In the first part, the distance of *zd* to the axis is constant and the resultant object is a torus. In the second part, this distance is varying and the resultant object is nonplanar.

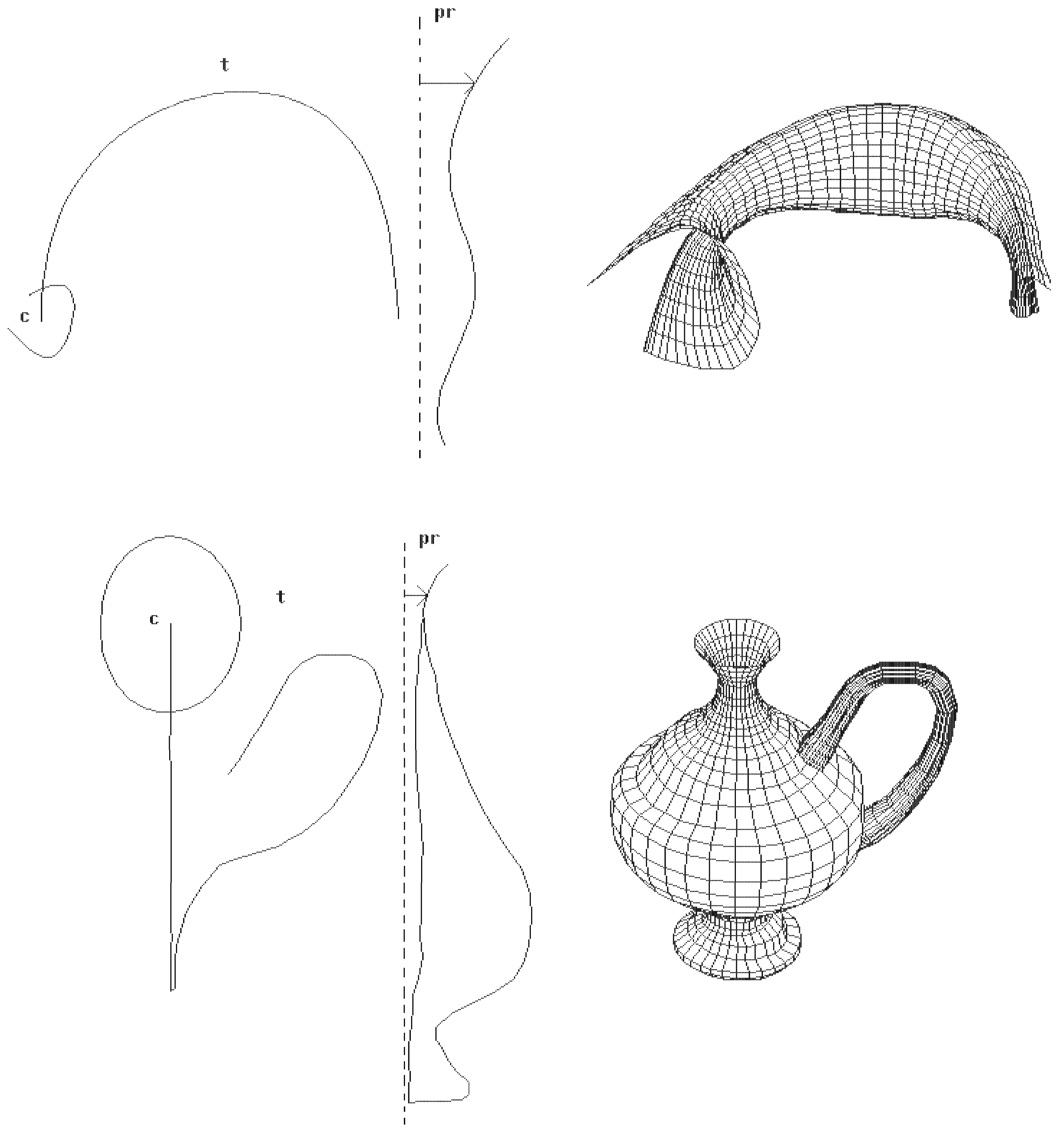


Figure 13 The contour curve is scaled to different sizes (via the curve *pr*) to obtain a profiled general sweep object.

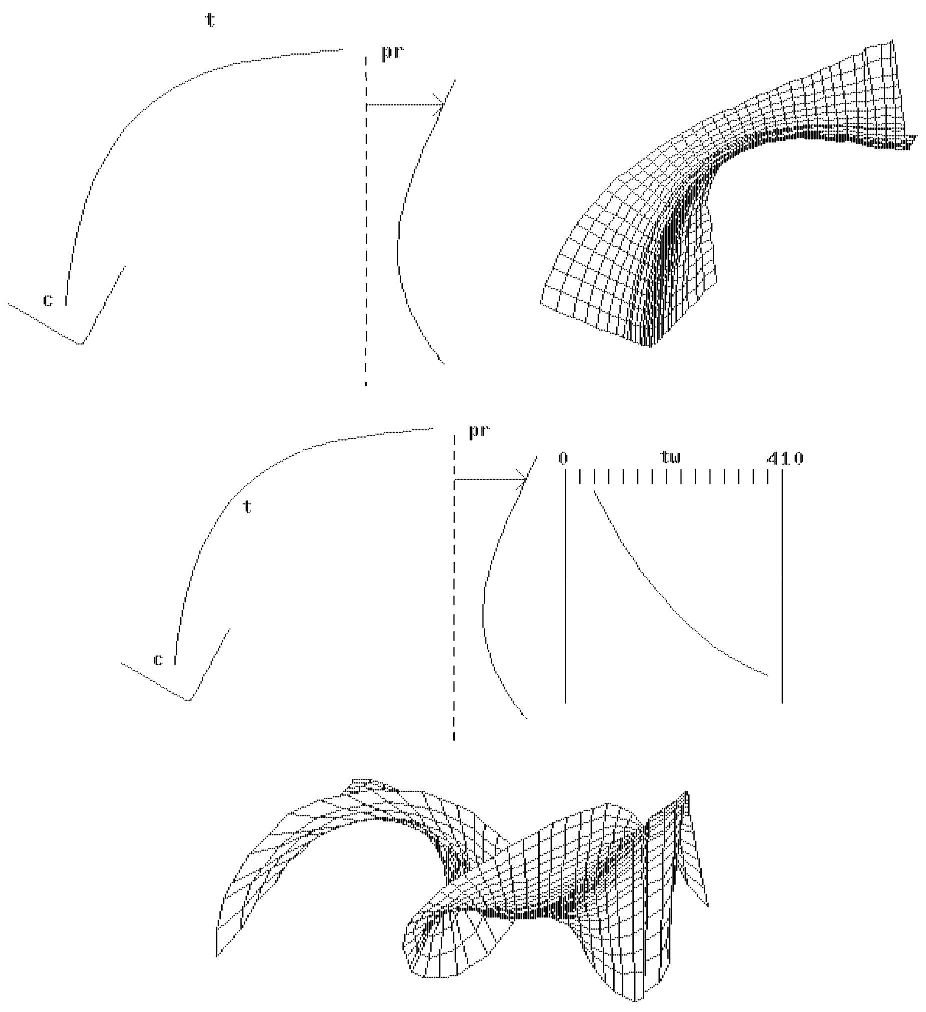


Figure 14 Contour *c*, trajectory *t*, profile *pr*, and twist *tw* curves and the produced objects. The first object is a profiled object. The second object is a twisted-profiled object.

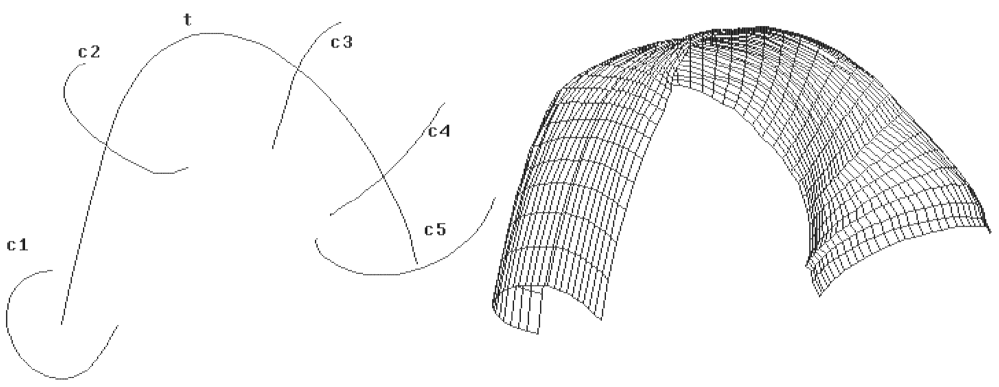


Figure 15 A trajectory curve and some contour curves with different shapes (c_1 to c_5) and the resultant sweep object with varying contour.

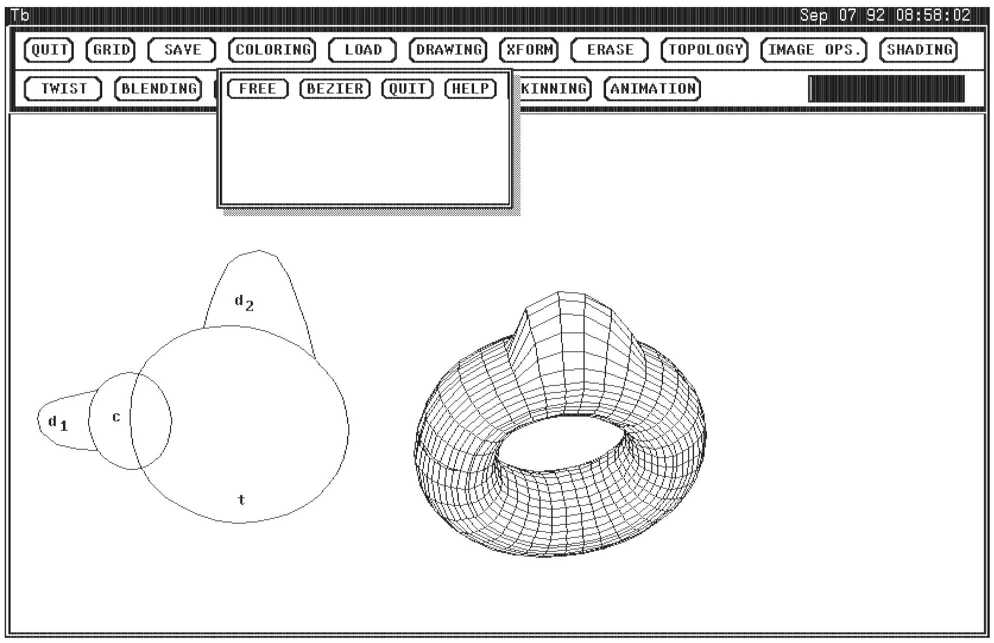
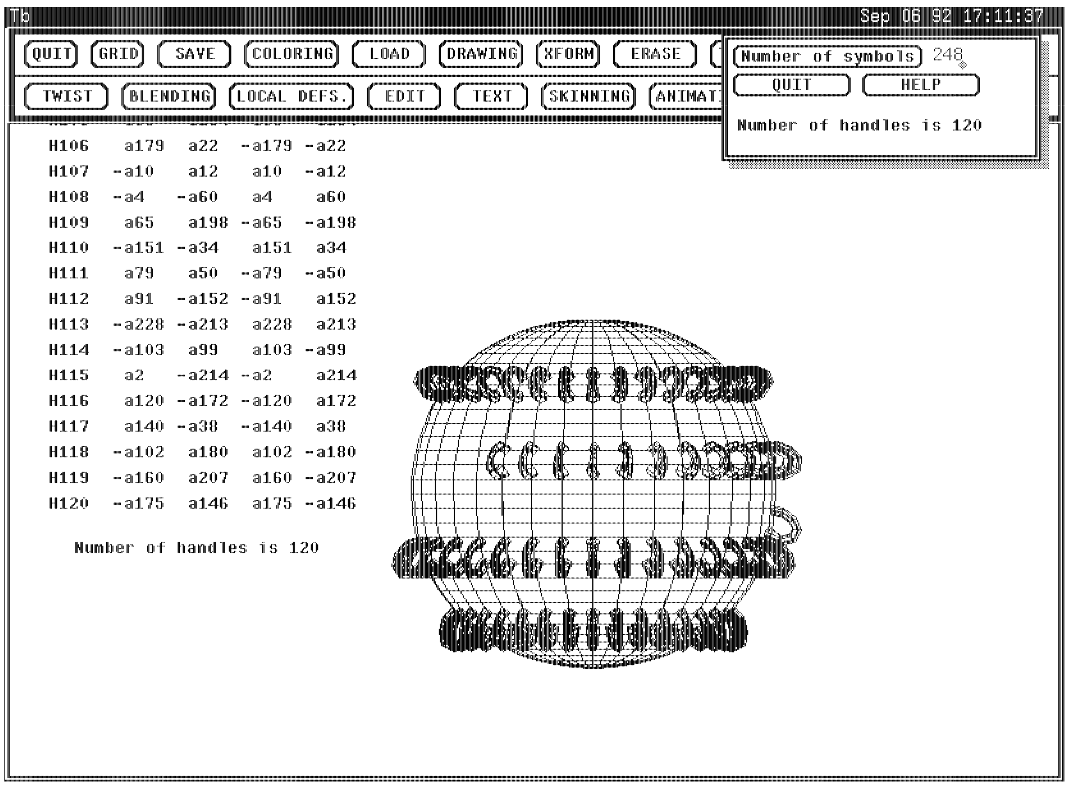


Figure 16 Local deformations on a sweep object. Tb locally deforms the object using d_1 and d_2 which are the first and the second local deformation curves, respectively.



```
Number of handles is 120
```

```
H106    a179  a22  -a179 -a22
H107   -a10   a12   a10  -a12
H108   -a4   -a60   a4    a60
H109    a65   a198 -a65  -a198
H110   -a151 -a34   a151  a34
H111    a79   a50  -a79  -a50
H112    a91  -a152 -a91   a152
H113   -a228 -a213  a228  a213
H114   -a103  a99   a103 -a99
H115    a2   -a214 -a2    a214
H116    a120 -a172 -a120  a172
H117    a140 -a38  -a140  a38
H118   -a102  a180  a102 -a180
H119   -a160  a207  a160 -a207
H120   -a175  a146  a175 -a146

   Number of handles is 120
```

Figure 17 A sphere with 120 handles. Symbols show the normalised handles. For example, H_2 a_1 a_6 $-a_1$ $-a_6$ denotes the second handle obtained by the symbols a_1 and a_6. (The minus sign is used here as a synonym for $^{-1}$.)

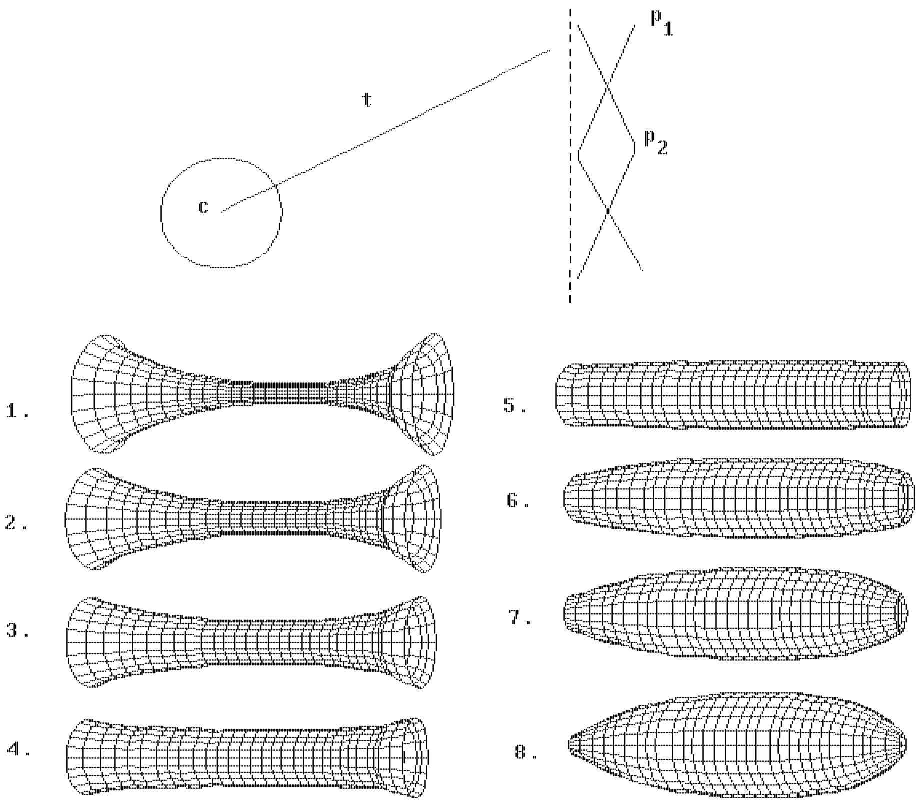


Figure 18 Animation by the help of the profile curve (c is the contour, t is the trajectory, and p_1 and p_2 are the first and the last forms of the profile, respectively). Numbers show the animation sequence.

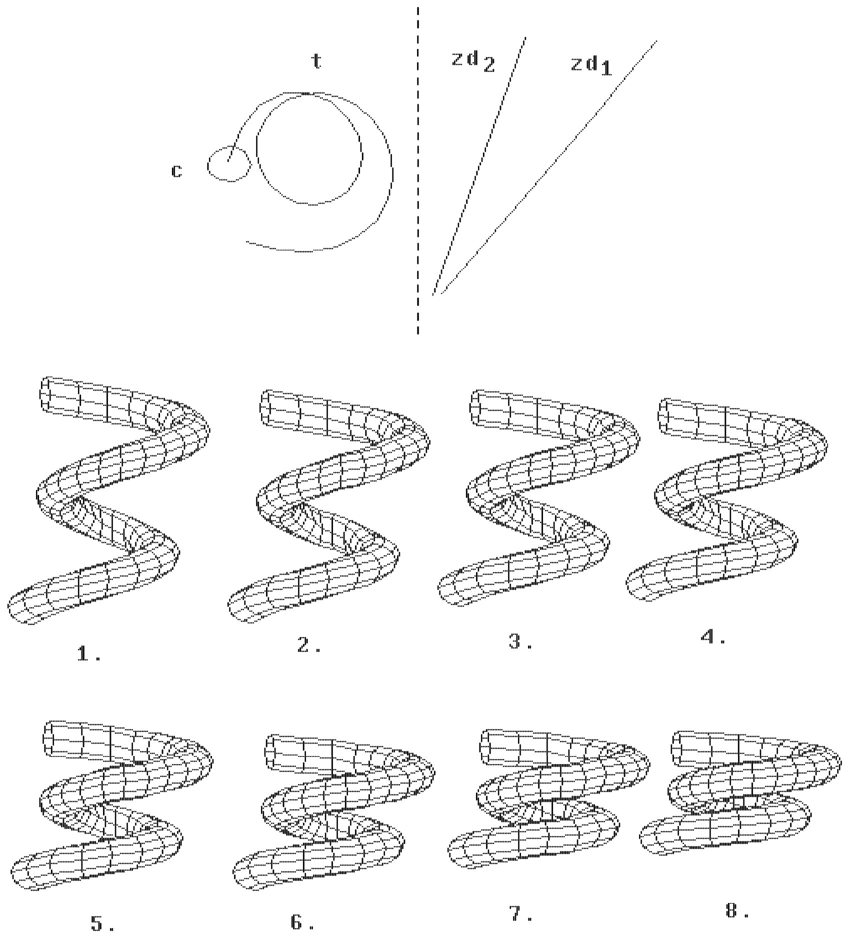


Figure 19 Animation by the help of the depth-modulation curve (c is the contour, t is the trajectory, and zd_1 and zd_2 are the first and the last forms of the depth-modulation curve, respectively). Numbers show the animation sequence.

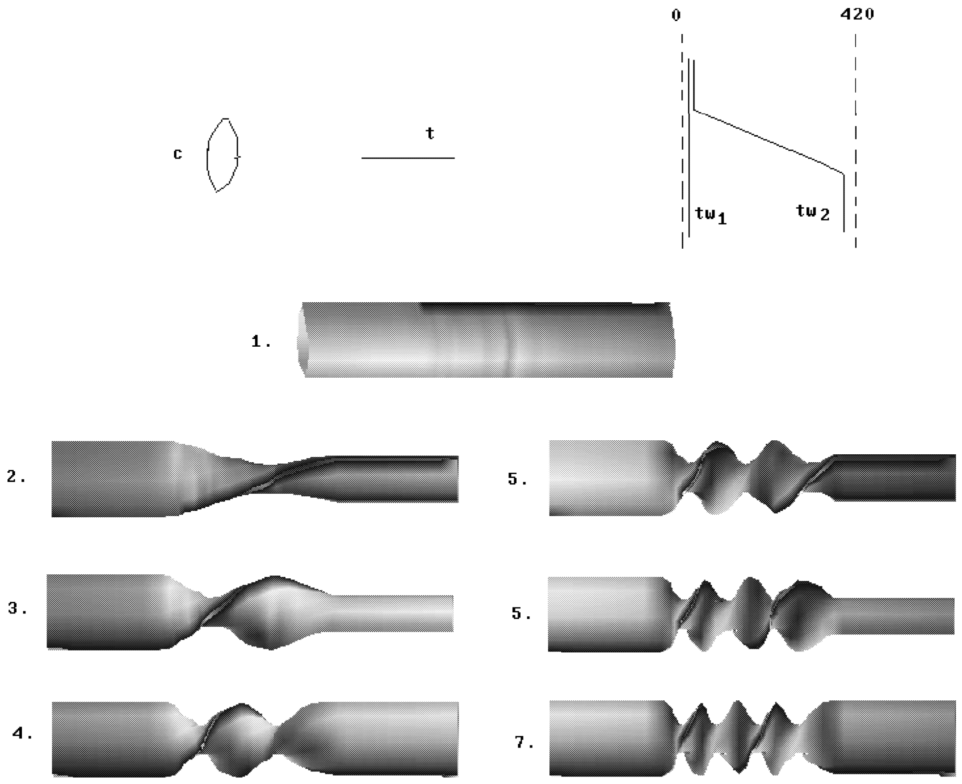


Figure 20 Local animation by the help of twist curve (c is the contour, t is the trajectory, and tw_1 and tw_2 are the first and the last forms of the twist curve, respectively). Numbers show the animation sequence.

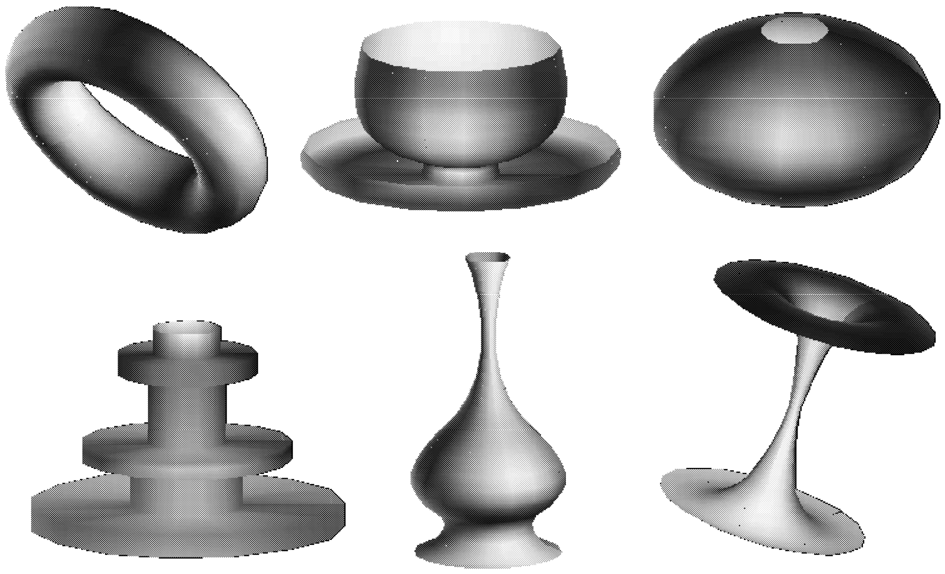


Figure 21 Some rotational sweep objects. The objects are represented with a contour curve and a radius in $\top\flat$.

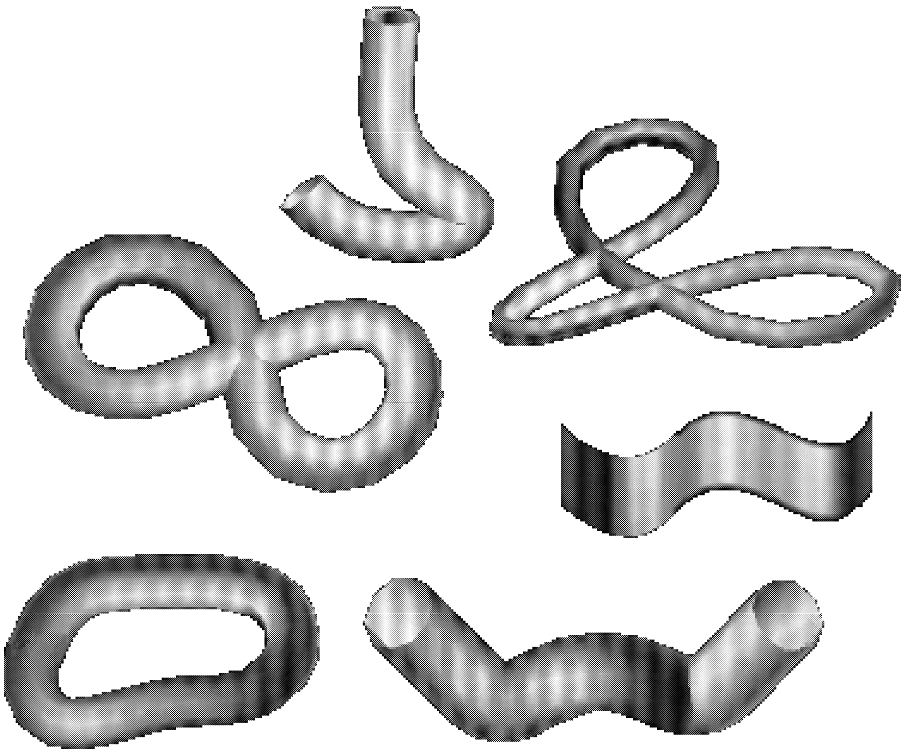


Figure 22 Some nonprofiled sweep objects. The objects are presented with a contour and a trajectory curve in $\top\flat$.

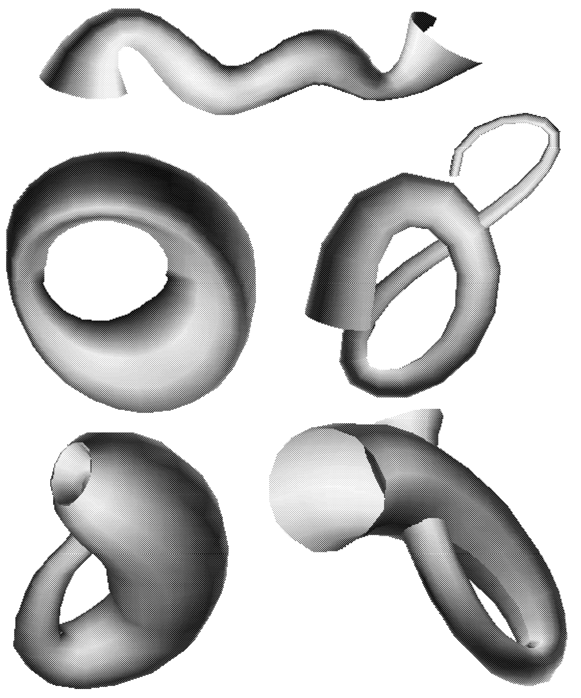


Figure 23 Some profiled sweep objects. The objects are represented with a contour, a trajectory, and profile curve in ⊤♭.

Figure 24 Some depth-modulated sweep objects. The objects are truly 3-dimensional and are represented with a contour, a trajectory, and a depth-modulation curve in ⊤♭. It is the depth-modulation curve that gives the third dimension to the trajectory.

Figure 25 Some twisted sweep objects. The objects are represented with a contour, a trajectory, and a twist curve in ⊤♭.

Figure 26 Some sweep objects deformed locally.

Figure 27 Some sweep objects with varying contours.

Figure 28 Transparent windows on a sweep object to inspect sublayers of the object.

Figure 29 Some objects produced by composing the sweep objects.

Figure 30 Some familiar objects produced by composing sweep objects.

Figure 31 A punctured surface of genus 4.

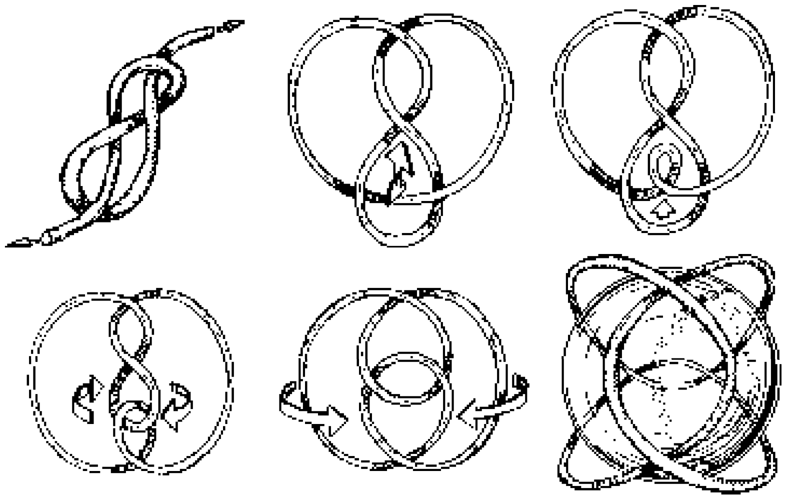


Figure 32 Projections of a knot (digitised from Francis' book (p. 150)).

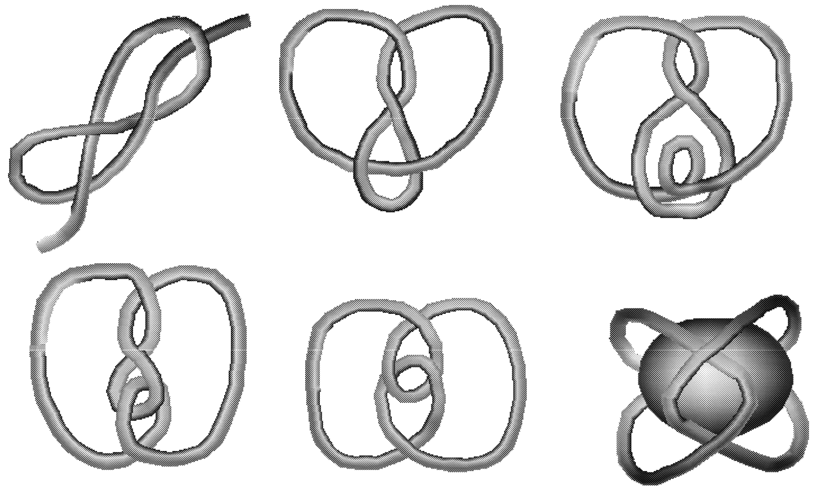


Figure 33 Projections of a knot. The picture was computed by ⊤♭.

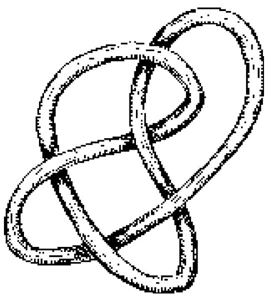


Figure 34 A knot (the first figure digitised from Francis' book (p. 38) and the second produced by T♭).

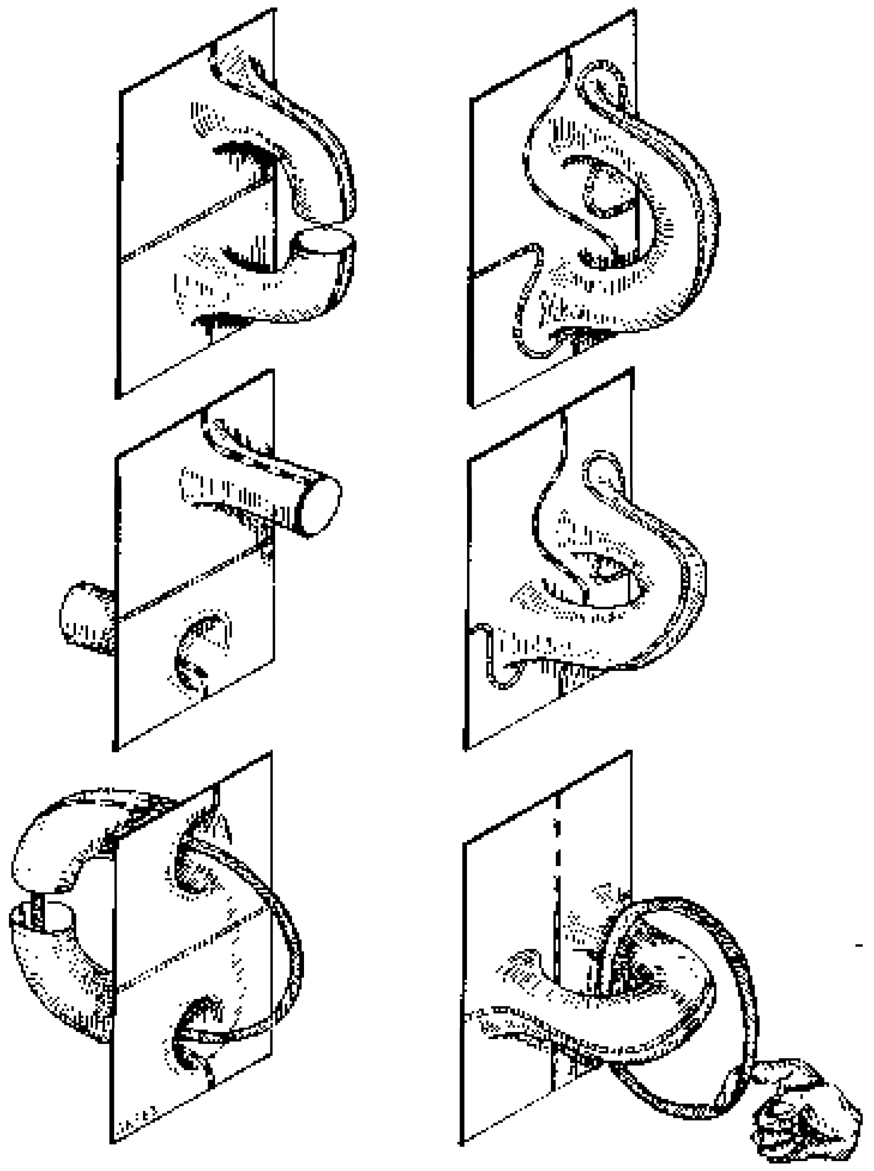


Figure 35 Swapping handle cores (digitised from Francis' book (p. 140)).

Figure 36 Swapping handle cores. This picture was computed by $\top\flat$.

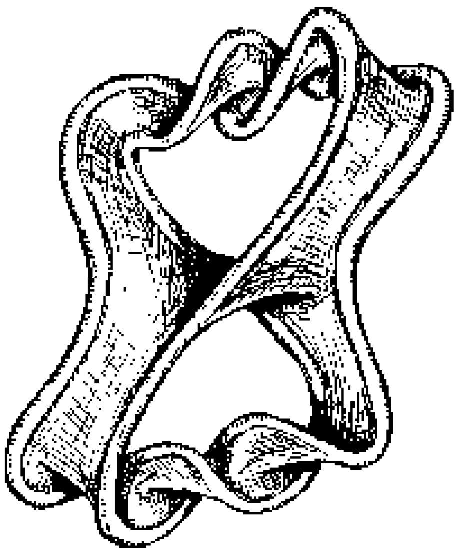


Figure 37 Eight knot (the first figure taken from Francis' book (p. 37) and the second produced by T♭).